Solving Some Differential Equations in Mathcad
(Prereq Differential Equations)

Indexing starts at zero

Forward-in-Time Calculations of Exponential Growth/Decay

$X_o := 1$ Initial value of Xo

$a := 1$ Growth Decay Factor (units of increase of X per time step)

$\frac{d}{dt}X(t) = a \cdot X(t)$ Relationship in Calculus Form (I'm using the bold comparative equals sign as an equation editor)

$F(X) := a \cdot X$ Here is a re-write of the above so I have the "forcings" that are used on X

$X_{true}(a,t) := X_o \cdot e^{a \cdot t}$ This is the true calculus-Dif-Eq solution

Now let's program this using the Euler's Method....
.... Here is the theory...

$$f[X(t_o)] = \frac{d}{dt}X(t) \approx \frac{\Delta X}{\Delta t}$$

$$f[X(t_o)] = \frac{d}{dt}X(t_o) \approx \frac{X(t_o + \Delta t) - X(t_o)}{\Delta t}$$

$$f[X(t_o)] \approx \frac{X(t_o + \Delta t) - X(t_o)}{\Delta t}$$

$$X(t_o + \Delta t) \approx X(t_o) + f[X(t_o)]\Delta t$$

... and some pseudocode...

```
x0 = 1.0
a  = 1.0
dt = 0.5

x = x0
loop i from 1 to 3 by 1
   rate = a * x
   x = x + rate * dt
end loop i

output(x)
```

And here I am making it work. I'm doing it a little differently, though so that we can put things in terms of our delta-X and make the code more flexible and PORTABLE (never reinvent a wheel if you don' t have to! ... unless it's for college credit... If you know what I mean*)

$X_{euler}(t_f, f_x, x_o, \Delta t) :=$
- $x_{output} \leftarrow x_o$
- $N \leftarrow \frac{t_f}{\Delta t}$ <- This line is new...
- if $N > 0$ <- This a "patch" for when N=0 that I added
  - for $i \in 1..N$
    - $x_{output} \leftarrow x_{output} + f_x(x_{output}) \cdot \Delta t$
- return x_{output}

<- Functions written above your "code" can be referenced by the code! This will be important in our root-finding code.

And now we let'r rip! $t_0 := 0$ $t_f := 1.5$ $\Delta t := 0.5$

$X_{euler}(t_f, F, X_o, \Delta t) = 3.375$

$X_{euler}(t_f, F, X_o, 0.25) = 3.815$

$X_{euler}(t_f, F, X_o, 0.05) = 4.322$

$X_{euler}(t_f, F, X_o, 0.00001) = 4.482$

And here is the real answer

$X_{true}(a, t_f) = 4.482$

So like the pi formula, the smaller the increment, the more crunching you need to do... but the more accurate the solution

* and I think you do...

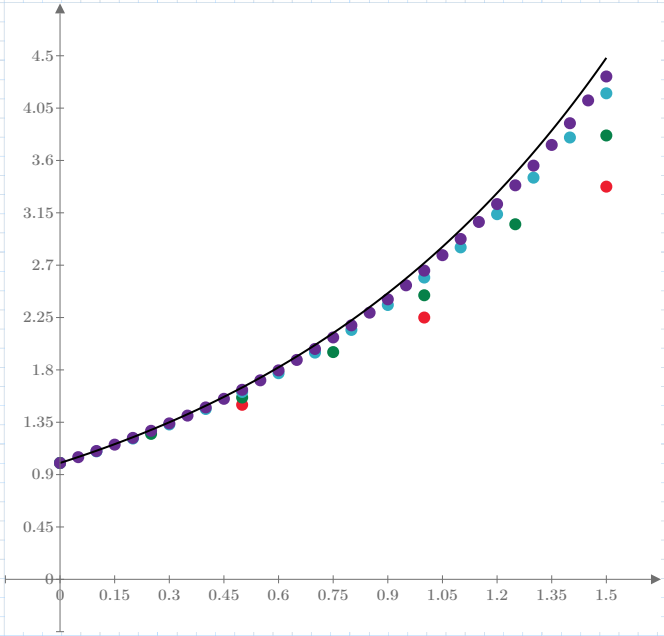
Solving Some Differential Equations in Mathcad
(Prereq Differential Equations)

Indexing starts at zero

We can also demonstrate this for a time series though Mathcad can be rather difficult when plotting it.

First let's create a generic time series for a delta-ts of from 0.5 and 0.05

$t_{0.50} := 0, 0.50 .. 1.50$ $t_{0.25} := 0, 0.25 .. 1.50$ $t_{0.10} := 0, 0.10 .. 1.50$ $t_{0.05} := 0, 0.05 .. 1.50$



$X_{euler}(t_{0.50}, F, X_o, 0.5)$

$X_{euler}(t_{0.25}, F, X_o, 0.25)$

$X_{euler}(t_{0.10}, F, X_o, 0.10)$

$X_{euler}(t_{0.05}, F, X_o, 0.05)$

$X_{true}(a, time)$

$t_{0.50}$

$t_{0.25}$

$t_{0.10}$

$t_{0.05}$

$time$

Your take-away point on this is as with the pi case, the smaller the increment over which you integrate a solution (we call marching forward in time "integrating [over time]"), the more accurate your result with be.

But this is at a price. It takes more steps to crunch through for a smaller delta-t. And that is an increase in expense especually when you are modeling large systems like watersheds and groundwater fields.

You CAN cut back on the number of timesteps AND retain accuracy. For those of you in Differential Equations there is a method called Runge Kutta that will achieve this..... ..... and with that people who are in or have had Dif-Eq's can move forward.

Solving Some Differential Equations in Mathcad
(Prereq Differential Equations)

Indexing starts at zero

...For those of you in Differential Equations...

There are some resources in Mathcad that help with solving differential equations.

Some can be used without solve blocks others need them...

Here is an example for the Solve Block.

Guess Values

$this\_space\_is\_blank := 0$

<- We're not using this but it's not as pretty if you just leave it blank (try deleting it and see)

Constraints

$\frac{d}{dt}X(t) = a \cdot X(t)$

$X(t_0) = 1$

<- Here we define a function X (not the same as what is on "the outside" of hte solve block).
<- We want to give it an initial condition

Solver

$X_{odesolve} := \text{odesolve}\left(X(t), t_f\right)$

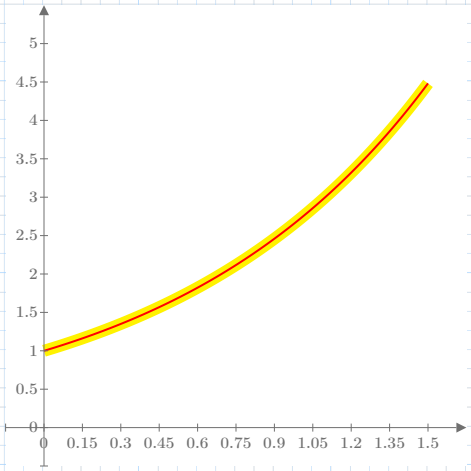
<- And this seals the deal with an ODE solver

Xodesolve is an abstraction kept deep in Mathcad with is a little frustrating... Your prof *hates* "black boxes."

$X_{odesolve} =$ "Function: <Closure_1>b__c"

$X_{odesolve} \rightarrow$ "Symbolic result is an invalid Prime expression"

You need to graph it to see



t

$X_{true}(a, t)$

$X_{odesolve}(t)$

The ODE is producing a solution that is valid between the time values requested in odesolve (0 and tf)

For a more discrete that will give you the output for each timestep. There is a function called rkfixed...

Solving Some Differential Equations in Mathcad
(Prereq Differential Equations)

Indexing starts at zero

The setup for this one is similar to what we did above with some small [passive agressively subtle] changes.

$t_0 := 0 \quad t_f := 1.5 \quad \Delta t := 0.5 \quad N := \frac{(t_f - t_0)}{\Delta t} = 3$

You need to have a start and end time and total number of time steps...

$f_{rk}(t, X) := a \cdot X$

You also need to state your function according to the value you're going to solve and the value overwhcih you integrate (whcih for us is "time")... these are the forcings that are beating on your value of X(t)

$X_{rk} := \text{rkfixed}(X_o, t_0, t_f, N, f_{rk})$

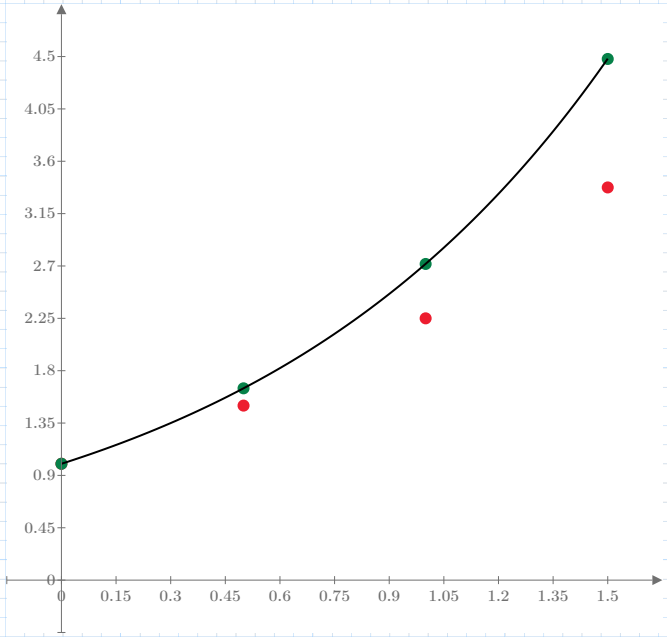
Then just provide the initial condition (for us, Xo)

$$X_{rk} = \begin{bmatrix} 0 & 1 \\ 0.5 & 1.648 \\ 1 & 2.717 \\ 1.5 & 4.479 \end{bmatrix}$$

The result is a time array in the "down" and across the resulting variables as listed your "function"

$time_{rk05} := X_{rk}^{\langle 0 \rangle} \quad X_{rkoft05} := X_{rk}^{\langle 1 \rangle}$

You can split them up with the <> operator



Now let's compare the two methods using the same time step of 0.5!

$X_{euler}(t_{0.50}, F, X_o, 0.5)$

$X_{rkoft05}$

$X_{true}(a, time)$

$t_{0.50}$

$time_{rk05}$

time